# Is Space a Stronger Resource than Time? Positive Answer for the Nondeterministic At-Least-Quadratic Time Case

Nicola Caporaso
CATTID, Università "Sapienza" di Roma
Piazzale Aldo Moro 5, 00185 Roma, Italy
Email: nicola.caporaso@uniroma1.it

January 26, 2010

**Abstract**

We show that all languages accepted in time $f(n) \geq n^2$ can be accepted in space $O(f(n)^{1/2})$ *and* in time $O(f(n))$. The proof is carried out by simulation, based on the idea of *guessing* the sequences of internal states of the simulated TM when entering certain critical cells, whose location is also guessed. Our method cannot be generalised easily to many-tapes TMs. And in no case can it be relativised.

## 1 Introduction

Let $T_M(n)$ and $S_M(n)$ denote the time and space consumed by a Turing Machine (TM ) $M$ which, given an input of length $n$, stops operating. Now, assume that $M$ is an acceptor for the language $L = L(M)$. From the *linear* space-compression theorem, for all constants $c$, one can find a new TM $M^*$ such that $L = L(M^*) = L(M)$ and

$$cS_{M^*}(n) \leq T_{M^*}(n) = T_M(n). \tag{1}$$

One might ask whether a better than linear result can be obtained. This is not a trivial question: after all, P$\stackrel{?}{=}$PSPACE is a major problem in computer science. The nondeterministic case is equally interesting, given that NP$\stackrel{?}{=}$NPSPACE is a major problem too.

We will prove the following

**Theorem 1** *For every* NTM $M$*, another* NTM $M^*$ *and a constant* $a$ *can be defined such that, for all input* $w$ *and* $n \geq |w|$*,* $M^*$ *accepts* $w$ *in time* $n^2$ *and space* $n$ *if and only if* $M$ *accepts* $w$ *in time* $an^2$*.*

This allows us to answer positively the question in the case of single-tape nonderministic TM (NTM ) at and above the quadratic time level. The following

**Corollary 2**

$$\text{ONE-TAPE-NTIME}(f(n)) = \text{ONE-TAPE-NTIMESPACE}(f(n); f(n)^{1/2})$$

is the main result of this paper.

We don't see an easy way to extend the result to many-tapes TMs. We would like to stress that this result cannot be extended to oracle-TMs either, because one cannot put an upper bound on queries to the oracle. One might speculate on the interest of such non-relativisable arguments in investigations on the separation problems.

The evaluation of the price (in terms of time) to be paid to save space is a topic of complexity theory that was initiated by Hopcroft and Ullman, who proved that deterministic and nondeterministic single-tape TMs respecting a time bound $T(n)$ can be simulated in space $T^{1/2}$ within a time exponential in $T(n)$ [1]. Ibarra and Moran [4] proved that single-tape TMs whose runtime is bounded above by $T(n)$ can be simulated in time $T(n)^{3/2}$ and space $T(n)^{1/2}$. As far as we know, however, *free-of-charge* results have not been proved so far. We show that not too long crossing sequences exist by a method that we have derived from from [1].

# 2 Definitions

We will introduce a NTM $M$ with a single half-infinite tape. The tape is partitioned in blocks, all except at most one of the same length $n$. The NTM will visit each block a certain number of times: we call each of these visits a phase. The sequence of all the visits $M$ makes on a given block is called the block's history. In the following section, we will see how $M^*$ works by trying to guess a possible story for the operation of $M$ until it arrives at the correct one.

Let us fix, for the remaining part of this paper, a NTM $M$, an input $w$ for $M$, and a number $n \geq |w|$. Let us identify the states of $M$ with the numbers $0, 1, \ldots$ Some states are deterministic, while others are not. Without any loss of generality (see for example [3], chapter 7) we may assume that

1. The tape is infinite to the right. We call *cell* $h$ the $h$-th cell ($h \geq 1$), counting from the left end of the tape. We use $\Delta$, often with affixes, as a variable defined on $\{-1, +1\}$. This variable will be used to identify the direction of motion of $M$ by understanding $-1$ to mean *left*, and $+1$ to mean *right*.

2. When in a deterministic state, $M$ either moves in the direction $\Delta$, or else it writes on its tape, *but not both*. If it tries to move left from cell 1, then it stops operating (but it may stop in other ways too). When in a nondeterministic state, it just chooses among a number $> 1$ of next states, *but it does not move or write*.

3. $M$ starts operating in the *initial* state 0, with $w$ stored in the cells $1, \ldots, |w|$. To accept, it tries to move left from cell 1 in the (only) *accepting* state 1.

We have a *computation* $C$ for each sequence of nondeterministic choices made by $M$ on $w$. The time for $C$ is the number of moves it includes, and its space is the number of distinct cells it visits. $M$ accepts its input within time $h$ and space $k$ if there is a computation that takes time $h$ and space $k$. Other computations may accept the input in time $h^* > h$ and/or space $k^* > k$, reject it, or never halt.

We will call $\beta_i$ the *boundary* between cells $i$ and $i+1$. We will focus on the behaviour of $M$ at evenly spaced boundaries, starting at $\beta_P$, with spacing $n$. Accordingly, for each $P \leq n$ we define a *partition* $\pi_P$ of the first $n^2$ cells into *blocks* in the following way: the block $B_1$ consists of the first $P$ cells, and $B_{j>1}$ consists of the $n$ cells from $P+(j-1)n+1$ to $P+jn$. We will call the boundary between two adjacent blocks $B_j$ and $B_{j+1}$, a *milestone* $\mu_j$; clearly, $\mu_j = \beta_{P+jn}$. In addition, we will call $\mu_0$ the left end of the tape.

For a given computation, let a *phase* denote the behaviour of $M$ during a single visit to a block, until it either stops operating without leaving the block, or it moves across a milestone. Phase 1 goes from the start to when $M$ leaves for the first time the block $B_1$ to enter, from the left, $B_2$. If by the end of phase $k$, $M$ leaves $B_j$ moving in the direction $\Delta$, then phase $k+1$ is the period of operation of $M$ on $B_{j+\Delta}$ until $M$ leaves it to come back to $B_j$, or to enter $B_{j+2\Delta}$.

A *descriptor* is a 4-ple $D = (p, j, i, \Delta)$ saying that, at the beginning of phase $p$, $M$ is in state $i$, and moves across $\mu_j$ in the direction $\Delta$. We adopt the following conventions:

1. We will sort descriptors by phase number into *sequences*. A sequence $L = J \oplus K$ is the result of composing sequences $J$ and $K$ by order of phase number.

2. The occurrence of a descriptor $D$ at places where one would expect a sentence means that $D$ is true w.r.t. the current computation $C$. Sequences of descriptors are truth-evaluated conjunctively. So, $L$ is *true/false* iff all/some of its elements are true/false. $J \to K$ means that if $J$ is true then $K$ is true.

Let us consider a computation $C$, consisting of $k$ phases, and a milestone $\mu_j$ ($j \geq 1$). Assume that $C$ goes for $m \geq 0$ times across $\mu_j$; then, its *history* $H_j$ is the sequence of descriptors of the form

$$H_j = (p_1, j, h_1, \Delta_1), \ldots, (p_m, j, h_m, \Delta_m) \tag{2}$$

where $\Delta_i$ is $+1$ if $i$ is odd and is $-1$ if $i$ is even (since a milestone is always first crossed from the left), and where $h_i$ is the state of $M$ when it crossed $\mu_j$ for the $i$-th time. The sequence is empty if $m = 0$. By definition, $H_0$ begins with $(1, 0, 0, +1)$, and it continues (and ends) with the descriptor $(p, 0, h, -1)$ iff $M$

$$
\begin{array}{ccccccccc}
\mu_0 & B_1 & \mu_1 & B_2 & \mu_2 & B_3 & \mu_3 & B_4 & \mu_4 \\
\rightarrow & & \rightarrow & & \rightarrow & & \rightarrow & & \\
 & & & & & & \leftarrow & & \\
 & & & & & & \rightarrow & & \\
\leftarrow & & \leftarrow & & \leftarrow & & \leftarrow & &
\end{array}
$$

$$
\begin{aligned}
H_0 &= (1,0,0,+1),(10,0,1,-1)\\
H_1 &= (2,1,i_1,+1),(9,1,i_8,-1)\\
H_2 &= (3,2,i_2,+1),(8,2,i_7,-1)\\
H_3 &= (4,3,i_3,+1),(5,3,i_4,-1),(6,3,i_5,+1),(7,3,i_6,-1)
\end{aligned}
$$

Figure 1: A typical history for the blocks of a TM $M$.

halts by trying to move left from cell 1. So, if $M$ accepts after $k$ phases, we have $H_0 = (1,0,0,+1),(k,0,1,-1)$. If $C$ visits $r$ blocks, then its *history* is the sequence

$$H = H_0, H_i, \ldots, H_{r+1} \tag{3}$$

where only $H_{r+1}$ is empty.

We call $H_j^+$ the sub-sequence of $\{H_j\}$ consisting of the descriptors ending with +1, and $H_j^-$ the sub-sequence of descriptors ending with -1. So we have $H_j = H_j^+ \oplus H_j^-$. The *in-history* $INH_j$ of block $B_j$ is $H_j^+ \oplus H_{j+1}^-$, and, symmetrically, its *out-history* $OUTH_j$ is $H_j^- \oplus H_{j+1}^+$. Finally, the *history of block* $B_j$ is given by $BH_j = INH_j \oplus OUTH_j$. A *story* $S$ is a guess on a history. Notations like $S_j, S_j^-, OUTS_j, \ldots$ and terms like *story about milestone* $\mu_j$, *out-story* of block $B_j$, etc. are the defined analogously to their *his*torical counterparts.

**Example** Assume that in a given computation $C$, $M$ moves right until $B_4$, then it oscillates twice between $B_4$ and $B_3$, and, finally, it goes left until cell 1 and accepts. This behaviour and the related histories are sketched out in Fig. 1.

# 3 Construction of M*

To determine whether a given accepting story coincides with a history, we need to introduce two NTM . The first one, called `phase`, takes as input an "incoming" and an "outgoing" descriptor, as well as a string, and attempts to simulate the operation of $M$ on a given block during a given phase. The second one, `check`, works on a block by iteratively calling `phase` and checking that a possible story of a block is coherent across all of its phases. Our NTM $M^*$ works by guesswork: it makes up a story for the whole tape (including how the tape is arranged in blocks), and calls `check` on all of the blocks to verify whether the

story is coherent. At the end of this section, we will show that the $M^*$ is able to guess the history correctly.

A NTM `phase` is employed to simulate the behaviour of $M$ during a phase on a generic block. It is so defined:

1. Given an input of the form

$$(p, j, i, \Delta), (p+1, j^*, i^*, \Delta^*), X,$$

   it starts operating in the state $i$ on a string of the form $\langle X \rangle$, immediately at the right of $\langle$ for $\Delta = +1$, or at the left of $\rangle$ for $\Delta = -1$. The symbols $\langle$ and $\rangle$ are not in the tape alphabet of $M$

2. The machine simulates faithfully the steps of $M$, so each nondeterministic choice made by $M$ causes (nondeterministically) different computations by `phase`.

3. `phase` stops the simulation if $M$ halts or if, after a left/right move by $M$, it scans $\langle$ or $\rangle$. Let $\langle X^* \rangle$ be the string produced by the current computation of `phase`. At this point, `phase` decides whether it will accept or reject.

4. `phase` rejects when one of the following conditions is verified: if it scans a symbol of $X^*$; if its state is not $i^*$; if $\langle$ is scanned, but $\Delta^* = +1$; and if $\rangle$ is scanned, but $\Delta * = -1$.

5. In all other cases `phase` accepts, and returns the string $X^*$.

Notice that different values for $X^*$ may be returned by the computations of `phase`.

**Lemma 3** *Assume that $D$ and $D^*$ are associated with phases $p$ and $p+1$, and that they occur respectively in the in- and out-histories of $B_j$; assume further that $X$ is stored in $B_j$ at the beginning of phase $p$. Then **phase** accepts and returns a content $X^*$ of $B_j$ iff we have $D \to D^*$.*

*Proof.* This follows immediately by construction of `phase`.

**Lemma 4** *Let a story be given. A* NTM ***check*** *can be defined which accepts $BS_j$ iff we have $INS_j \to OUTS_j$*

*Proof.* The initial content $X(j,0)$ of $X_j$ consists of a string of $n$ zeroes if $j > 2$. In $X(1,0)$ we find either the first $P$ symbols of $w$ if $P < |w|$, or $w$ followed by $P - |w|$ 0s. $X(2,0)$ begins with the part of $w$ not stored in $B_1$ (if any), followed by a string of zeroes.

NTM `check` works by iterating calls to NTM `phase` :

1. `check` calls `phase` with the following input: the $(2p-1)$-th and $2p$-th descriptors of $BS_j$, and $X(j, p-1)$.

2. If `phase` rejects, then `check` rejects too, and stops operating.

3. If `phase` accepts and returns $X^*$, `check` puts $X(j,p) = X^*$ and starts the $(p+1)$-th repetition.

If the last repetition of `phase` accepts, then `check` also accepts; else, it rejects. This ends the definition of `check`.

We are now in the position to define the NTM $M^*$. Assume that $M$ accepts. Our NTM will simulate $M$ as follows:

1. $M^*$ produces a guess for the values of the time $n^2$, the length $P$ of the first block, the total number of visited blocks $r$, and the number of phases $k$.

2. $M^*$ produces a guess for an accepting story $S = S_0, \dots, S_{r+1}$. Since $S$ is accepting we have $S_0 = ((1,1,0,1),(k,1,1,-1))$ and $S_{r+1}$ is empty.

3. Next, $M^*$ calls `check` $r$ times with input $BS_j$ $(1 \le j \le r)$.

4. If any call to `check` rejects, then $M^*$ rejects too; otherwise, $M^*$ accepts.

**Lemma 5** *If all calls to `check` accept, then $S$ is an accepting history.*

*Proof.* From lemma 4 and from the hypothesis of this lemma, the following implications are all true

$$\begin{array}{l} S_0^+ \wedge S_1^- \to S_0^- \wedge S_1^+ \\ S_1^+ \wedge S_2^- \to S_1^- \wedge S_2^+ \\ \dots \\ S_j^+ \wedge S_{j+1}^- \to S_j^- \wedge S_{j+1}^+ \\ \dots \\ S_r^+ \wedge S_{r+1}^- \to S_r^- \wedge S_{r+1}^+ \end{array} \tag{4}$$

Now, note that each $S_j^-$ $(1 \le j \le r)$ occurs in the antecedent of the $j$-th implication and in the succedent of the $(j+1)$-th implication; while each $S_j^+$ occurs in the succedent of the $j$-th, and in the antecedent of $(j+1)$-th one. Thus all $S_j^\pm$ can be eliminated. Note further that $S_{r+1}^-$ and $S_{r+1}^+$ are absent (empty). Thus the above reduces to $S_0^+ \to S_0^-$. Since $S_0^+ = (1,1,0,+1)$ is true by definition, we have that $S_0^- = (k,1,1,-1)$ is true. Hence, since all its descriptors are true, $S$ is an accepting history.

BV

# 4 Complexity

Let us begin by analysing the space required by $M^*$. This NTM needs space for two activities: simulations employing `phase`, and storing the story $S$. The former works on blocks, so it clearly requires $O(n)$. Since $S$ consists of $k$ descriptors, and the length of each of them is $\le c$, for a constant $c$ depending on $M$, we have $|S| \le ck$. The part of the theorem regarding space follows from the next lemma, and the fact that $k \le n$ implies that $|S|$ is also $O(n)$.

**Lemma 6** *For each accepting computation $C$ there is a partition $\pi_P$ such that its history consists of $k \leq n$ phases.*

*Proof.* Ad absurdum. Assume that for all $P$ we had a number of phases $k(P)$ such that $k(P) > n$. Since the overall number of moves across all boundaries is $\leq n^2$, and since each boundary is a milestone for precisely one partition $\pi_P$, we would have

$$\sum_{P \leq n} k(P) \leq n^2. \tag{5}$$

However the hypothesis ad abs. says that for each $\pi_P$ we have $k(P) > n$; that is

$$\sum_{P \leq n} k(P) > n^2. \tag{6}$$

This proves the lemma.

We conclude the proof of theorem 1 by analysing the time employed by $M^*$:

1. Time for the guesses is obviously linear.

2. By storing in the finite control of `phase` a description of $M$, we may arrange that it takes a time linear in the time spent by $M$ (a constant number of moves for each simulation of a step by $M$). Since each phase is simulated once, the overall time consumed by all calls to `phase` is $O(n^2)$.

3. We have to add a time $O(n)$ for the $r \leq n$ calls by $M^*$ to `check` and $O(n)$ for the $k$ calls by `check` to `call`.

By summing up these amounts, we obtain a time $an^2$, for some constant $a$ depending on the NTM $M$.

# References


[1] Hopcroft, J. E., and Ullman, J.D.: *Relations between time and tape complexities* in J. Assoc. Comp. Mach. 15(1968) 414–427.

[2] Hopcroft, J. E., Paul, W.J. and Valiant, L.G.: *On time vs. space* in J. Assoc. Comp. Mach. 24.2(1977) 332–337.

[3] Hopcroft, J. E., and Ullman, J.D.: *Introduction to automata theory, languages and computations.* Addison-Wesley, 1979.

[4] Ibarra, O.H. and Moran,S.: *Some time-space tradeoffs concerning single-tape and offline TMs.* SIAM J. Computing, 12.2(1983) 388–394.